89-17

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
		Reprint

4. TITLE AND SUBTITLE
Virtual Manufacturing

5. FUNDING NUMBERS
DAAL03-87-K-0005

6. AUTHOR(S)
Pentland, Alex P. and Williams, John R.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Massachusetts Institute of Technology CCRE/PACT
77 Massachusetts Avenue Room 1-175
Cambridge, MA 02139

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U. S. Army Research Office
P. O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ARO 24620.21-EG-UIR

11. SUPPLEMENTARY NOTES
The view, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The role of computers in the mechanical design process is currently limited to drafting and, if expert systems prove useful, helping routine design detailing. Their great potential for analysis is largely limited to checking single pieces of already-copleted designs. The most glaring defect in this situation is that the critical early stages of design--where different designs are compared and the general outlines of the final design are fixed--are conducted without computer support. This research argues that computers can fundamentally change and improve the design process by allowir interactive explortation of the space of valid deisgns. Computers can accomplish this by providing a drawing medium whose agility is comparable to epncil and paper, and which allows real-time structural and dynamic analysis of the entire system as it is being designed. There are several fundamental computational probelms in achieving this goal. We describe a solution that depends upon a basic change in how we rpresent and analyze structures, and describe a prototype syste which has demonstrated significant promise.

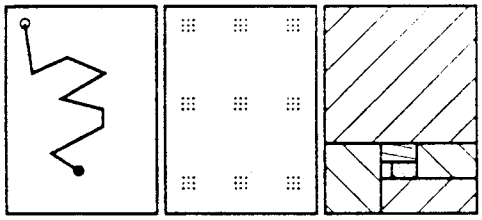
19950314 090

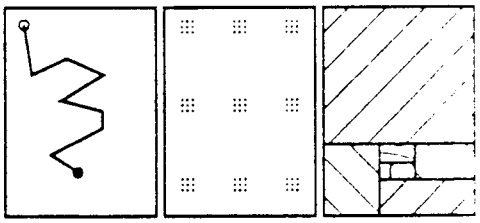
14. SUBJECT TERMS	15. NUMBER OF PAGES
Virtual Construction; Virtual Manufacturing; Simulation	16. PRICE CODE

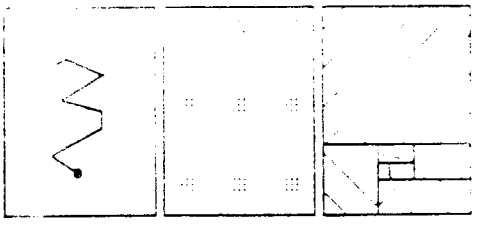
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Preprints

NSF Engineering Design Research Conference







College of Engineering
University of Massachusetts
Amherst

June 11 - 14, 1989

VIRTUAL MANUFACTURING

Alex Pentland and John Williams [1]

Vision Sciences Group, The Media Lab, and
Civil Engineering Department
Massachusetts Institute of Technology

ABSTRACT

The role of computers in the mechanical design process is currently limited to drafting and, if expert systems prove useful, helping routine design detailing. Their great potential for analysis is largely limited to checking single pieces of already-completed designs. The most glaring defect in this situation is that the critical early stages of design — where different designs are compared and the general outlines of the final design are fixed — are conducted without computer support. We argue that computers can fundamentally change and improve the design process by allowing *interactive* exploration of the space of *valid* designs. Computers can accomplish this by providing a drawing medium whose agility is comparable to pencil and paper, and which allows real-time structural and dynamic analysis of the entire system as it is being designed. There are several fundamental computational problems in achieving this goal. We describe a solution that depends upon a basic changes in how we represent and analyze structures, and describe a prototype system which has demonstrated significant promise.

1 INTRODUCTION

One can divide mechanical design into two very different categories: routine design, where the designer simply picks a design from among a family of designs that is already familiar, and novel design, where the designer confronts a new problem and must search for novel solutions. The area of routine design has been the focus of most design research because it can be described by a set of standard rules and thus is susceptible to expert system technology and to standard computational techniques.

We are not concerning ourselves with routine design. The real potential for advances, as we see it, is in the area of novel design. An analogy with the semiconductor industry will help to make this clear. When chips were very expensive to design, people normally adapted dozens (or hundreds) of standard chips to solve their particular problem. The result, of course, was often complex, unreliable, and expensive. Now that automatic tools for simulating, verifying and designing chips are becoming available, people are demanding custom chips that are designed from scratch to optimally solve their problem. Similarly we anticipate that routine design will become much less common if designers can cheaply and quickly optimize a design to address a particular situation.

[1]This research was made possible by National Science Foundation Grant No. IRI-87-19920 and by ARO Grant No. DAAL03-87-K-0005

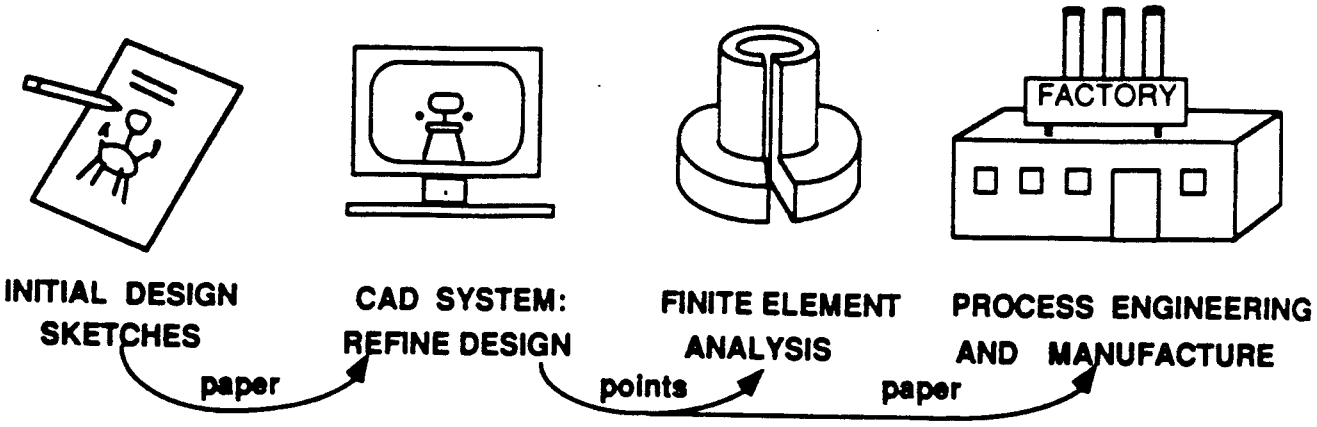


Figure 1: The design and manufacturing process today.

Our aim, therefore, is to cause a similar change in mechanical and structural design. We hope to accomplish this by changing the *medium* of design: by providing a simulated world within the computer in which the designer can explore and evaluate new designs in much the same way we today build and test scale models, but with the agility, immediacy and convenience of pencil and paper. In short, we propose to integrate computer analysis into the initial, exploratory phases of design.

Because we are interested in novel design, current expert systems technology will not help us very much. We believe that, just as was the case with chip design, progress depends more on having interactive design tools that allow quick design verification rather than on having design advisors based on fixed rules. Progress depends upon giving the designer convenient drawing and analysis tools rather than on replacing him.

At its core this paper is about solving some very hard computational problems that must be addressed before we can hope to automatically analyze large dynamical systems quickly enough to guide — as opposed to merely check — the design process. To get to these problems this paper first starts with a critique the design process as it is today, and then proposes a different model of the design process: *Virtual Manufacturing*. The paper then examines the fundamental computational problems that prevent the efficient, automatic analysis of large dynamic systems required by the Virtual Manufacturing paradigm, and finally we will describe the our proposed solution to these problems.

2 DESIGN TODAY

Figure 1 illustrates the four stages of design typically encountered in today's large engineering companies. The first stage of design is perhaps the most important, as it is where the design is roughed out and the basic directions, trade-offs, and goals of the design are decided. The senior designer starts with pencil and paper, a calculator, reference books, and his experience, and proceeds to sketch a series of design alternatives. Following each sketched design comes a rough evaluation of the design in terms of factors such as aesthetics, cost, strength, and manufacturability. This evaluation is then used to guide creation of the next design sketch.

The second stage is drafting. First the rough initial design is translated into the computer's language of points, lines, and surfaces, and then more junior designers flesh out the details of the design. It is this stage at which most computer aided design (CAD) and expert system tools are aimed.

The third stage is analysis. In this stage the each part of the finished design is translated into a finite element grid, and *separately* subjected to various analyses. Because finite element techniques are computationally expensive and can applied only to single bodies, analysis of a complex structure often requires that an entirely new, simplified computer model be generated.

The fourth stage is manufacture. Here a geometric specification (e.g., blueprints) are given to a process engineer, who is in charge of designing the manufacturing process. The blueprints are interpreted by use of convention: both the design and process engineer share knowledge of manufacturing techniques, materials, etc., which allows them to anticipate/guess at the other's problems and intentions.

To an extent these four stages are a caricature, and often things aren't *really* quite this bad – especially in smaller, more innovative companies. However we believe that the basic elements are correct, and that most proposals do little to address many of the fundamental problems.

2.1 Today's Problems

The most striking aspects of this process are that:

- The important first stage, where the basic design is established, is largely unchanged since the year 1900.
- Detailed analysis is only done post-hoc, long after the basic design has been established. As a consequence design flaws are often addressed by quick patches, rather than by fixing the fundamental problem.
- Complex systems can't really be analyzed; you have to custom-build a simplified model or analyze single, isolated pieces.
- Communication between the design, analysis and manufacturing stages depends overwhelmingly on use of shared knowledge, including implicit assumptions about techniques, approaches and materials. The fact that shared assumptions are so important makes it very difficult to introduce new techniques or materials.

3 VIRTUAL MANUFACTURING

Now we will propose a different way of doing design. To illustrate, we will step through a simple example — designing a chair — using our design system. In this example timings for each simulation task assume use of a workstations with roughly ten MFlop performance; the values presented were obtained by dividing times obtained on a Symbolics 3600 by ten. Because our current system is only a prototype, not all of the necessary functionality has been implemented so that timings reported are for analysis of a similar dynamic systems containing only currently implemented interactions. User interface timings are as actually measured on a Symbolics 3600; these timings were *not* divided by ten or otherwise scaled.

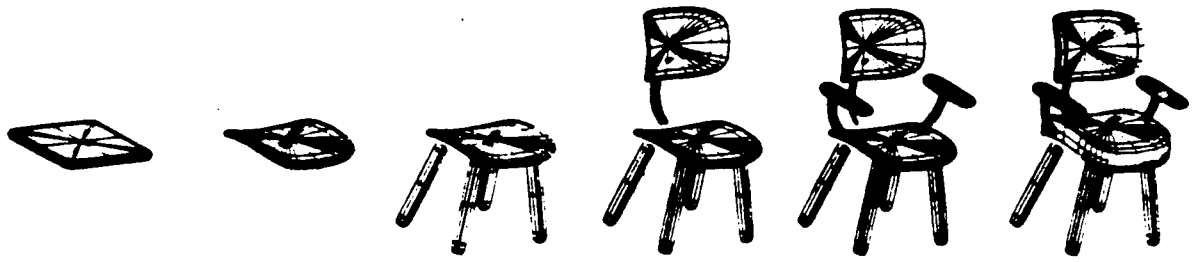


Figure 2: (a) The designer creates a sheet of material, (b) shapes it into a seat, and (c)-(f) adds other parts to form a chair. Elapsed time: 10 minutes.

3.1 Designing A Chair by Virtual Manufacture

Even for so simple a task a designing a chair, the designer faces a complex problem. Each element of the design affects many other elements, so that the entire design must be considered at once. For instance, the chair frame's material, the the thickness of the legs and other structural members, or the arrangement of structural members can all be varied in order to achieve the functional goal of a sturdy chair. These same variables must simultaneously be optimized to achieve an inexpensive chair, a manufacturable and long-lasting chair, and an attractive chair. Thus our designer has to optimize over many parameters to achieve a strong, economical, manufacturable, and attractive design.

How is our designer to proceed? Let us imagine ourselves in his place, sitting down to engage in a little Virtual Manufacturing. We are presented with engineering-style views of an empty space. The user interface, a system called SuperSketch [1,2], allows us to create objects (e.g., beams, pipes, blocks, etc.) made of particular materials, and then to interactively shape that object using a variety of manufacturing/forming operations. These shaped objects can then be attached to each other, and relations between them (that they are the same size, that they are parallel, etc.) can be established by use of constraints. This process is illustrated in Figure 2.

The system is extremely agile: objects move, change shape, and attach themselves in real time. On our current system (running on a Symbolics 3600) we have held races between people using clay and people using SuperSketch to see who can build an object faster: it usually ends in a tie [2]. The difference, of course, is that the person using the computer is left with an accurate 3-D CAD model rather than just a lump of clay.

So to design a chair we start by creating a sheet of some specific material (steel, for instance), and then shape it into a seat by bending it. We then create the chair back in a similar manner, and form arms, legs and attachments by bending and joining solid (or tubular) metal stock. Finally we add cushions shaped to fit the seat and back, thus obtaining the structure shown in Figure 2. Elapsed time from start to finished chair: 10

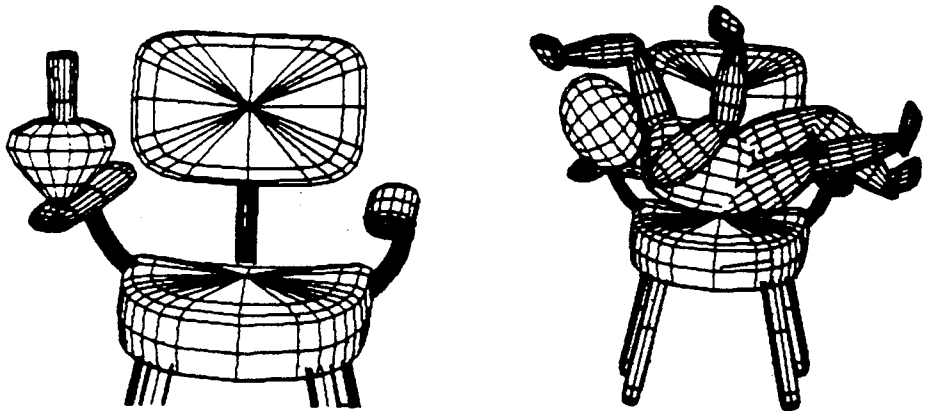


Figure 3: (a) Testing static loading of the chair (15 seconds), (b) testing dynamic loading of the chair (2.5 minutes)

minutes.

We now make use of the analysis capabilities of the system: we place static loads on the seat, arms, or chair back and then "turn on physics." The chair promptly sags a bit, as is illustrated in Figure 3(a). Elapsed time: 15 seconds (assuming 100 time steps of dynamic simulation are required to reach a static equilibrium).

Following these tests we might strengthen the attachment members, so that they won't bend as much. We might then turn on physics again, and see that an acceptable shape is maintained. Elapsed time to change attachment member diameter: 1 minute. Elapsed time to retest design: 15 seconds.

Finally, we might test the chair for dynamic properties: For instance, how does the chair hold up when someone throws themselves in it from three feet up? We can accomplish this simulation by dropping a simple body model from a height of three feet, as shown in Figure 3(b), and observing the result. Elapsed time for dynamic test: two and one-half minutes (assuming 1000 time steps of dynamic simulation).

3.2 Design Process Changed

This approach to design has goals that are quite different from those of most design research projects. Our system is *not* intended to perform any of the designer's functions, or even offer advice. It simply gives the designer a better *design medium*. By shortening the time between idea, realization, and analysis the designer can better explore the space of *physically valid designs*, and thus can better optimize the design.

The close integration of manufacturing, analysis and design that is implied by the phrase "Virtual Manufacturing " considerably simplifies the the design process illustrated in Figure 1. The main change is that we have collapsed the analysis stage into the first two stages of design. This allows us to *automatically* record how successive analyses caused changes in design, and thus leaves us with a set of structural tests that any subsequent change in the design must still satisfy. In other words, by moving the design-analysis-redesign loop

from inside the designer's head to inside the computer we can begin to record the design's *semantics* as well as its geometry.

Perhaps equally important as capturing design semantics is that by providing design tools that work (approximately) like the manufacturing processes used on the factory floor we can foster a natural integration of functional/geometric and manufacturing/process planning. For instance, if the designer selected steel tubing for the chair arms the system should provide only those forming operations that are plausible for steel tubes. By giving the designer virtual tools that correspond (at least roughly) with the available real tools, the designer ends up roughing out the manufacturing process at the same time as the geometry.

4 FUNDAMENTAL PROBLEMS

To some people these ideas sound old hat. To an extent this is true: some of these same ideas were behind both Sketchpad [3] and Thinglab [4], and there are now even start-up companies that offer constraints, finite element analysis, and interactivity. However nobody offers a system that is even close to being as agile and convenient as pencil and paper, and so initial design (as opposed to detailing) isn't done on a computer. And finite element analysis is so expensive and time-consuming that – even given desktop supercomputers – analysis is staying separate from design, except for special one- and two-dimensional cases involving a single body.

The reason no one has achieved really interactive "simulated manufacturing" or interactive 3-D, multibody analysis is that it is just too computationally expensive — at least when using standard techniques. Nor is it a matter of just having faster computers, because it is the *scaling* of these problems that kills things. That is, the cost escalates so quickly with increasing problem size that even the fastest supercomputers cannot solve realistic problems sufficiently quickly. To achieve the analysis and manipulation performance required by the Virtual Manufacturing scenario, therefore, requires addressing fundamental problems of computational complexity.

It appears to us that most of these complexity problems stem from representing geometry in terms of a collection of points rather than in a more analytic manner. For instance, in the above example there are roughly $p = 72$ points per part, a total of $n = 216$ unknowns (one unknown for each point's x, y, and z). Each time step of a dynamic simulation requires solving for each of these unknowns. The number of computer operations required in this simplest of cases is kn^2 (for $n = 216$ roughly $46000k$) using an explicit solution technique or somewhat more when using an implicit (direct) solution technique.[2] That's rougly three seconds of computation per time step on the high-performance workstation assumed in the above Virtual Manufacturing example. Similarly, to detect and characterize contact between m of these simple shapes requires $kp^2 m \log m$ operations, about four seconds of computation for this simple system of eleven interacting solids. So to simulate the system shown in Figure 3 for the 100 time steps required to do a static loading takes almost 12 minutes, rather than 15 seconds.

[2]Implicit (direct) solution techniques are often used because their stability allows large time steps even though each time step is more expensive. The trade-offs between our system, standard explicit techniques, and implicit techniques are discussed in the appendix.

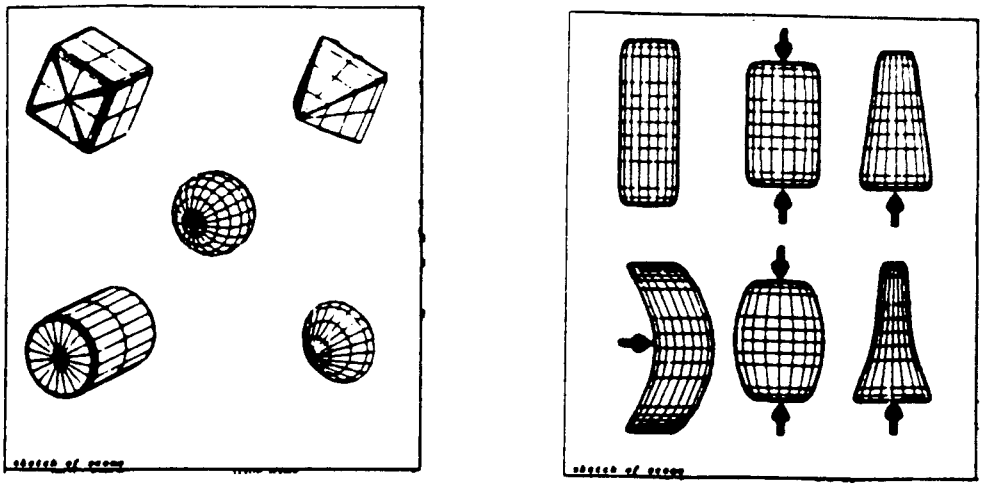


Figure 4: (a) Superquadrics: cube, $\epsilon_1 = 0.05, \epsilon_2 = 0.05$; cylinder, $\epsilon_1 = 1.0, \epsilon_2 = 0.05$; sphere, $\epsilon_1 = 1.0, \epsilon_2 = 1.0$; cone, $\epsilon_1 = 1.0, \epsilon_2 = 2.0$; pyramid, $\epsilon_1 = 2.0, \epsilon_2 = 2.0$. (b) Superquadrics subjected to various forces, in order to illustrate the appearance of the first and second strain modes. Arrows indicate applied forces. Top row: undeformed superquadric and two superquadrics subject to first-order (linear) strain modes; bottom row: three superquadrics subject to both first and second order strain modes.

You can see that we won't explore much of the design space at that rate. What is worse is what happens as we get away from small, simple systems, since the computation scales as n^2 and $p^2 m \log m$, which means that we can't conveniently handle even fairly modest systems even if we *do* use supercomputers. It is this scaling, which stems from using point representations of the object, that prevents achieving the capabilities required in the Virtual Manufacturing scenario.

5 A NEW APPROACH

To escape the vicious scaling problem of point-wise representation we need a type of model that (1) allows easy detection and characterization of object-to-object contact, and (2) can simply and analytically represent the surface as it deforms under stress. One example of such a type of model is the class of superquadrics taken together with a set of parameterized global deformations that correspond to the object's strain modes.

5.1 Volumetric Modeling Primitives: CSG and Dynamics

Superquadrics are a family of parametric shapes first investigated by the Danish designer Peit Hein [5]. They are a generalization of the equation of an ellipsoid $((x/a_1)^2 + (y/a_2)^2 + (z/a_3)^2 = 1)$; however rather than using only powers of two, a superquadric allows fractional powers:

$$((x/a_1)^{2/\epsilon_1} + (y/a_2)^{2/\epsilon_1})^{\epsilon_1/\epsilon_2} + (z/a_3)^{2/\epsilon_2} = 1. \quad (1)$$

The variables x and y are grouped together to allow separate control of the $x-y$ profile and the $z-(xy)$ profile via the shape parameters ϵ_1 and ϵ_2. The shapes generated by various values of ϵ_1 and ϵ_2 are shown in Figure 4(a).

A version of Equation 1 that is parameterized by generalized latitude η and longitude ω is:

$$\vec{X}(\eta,\omega) = \begin{pmatrix} a_1 \cos^{\epsilon_1}(\eta)\cos^{\epsilon_2}(\omega) \\ a_2 \cos^{\epsilon_1}(\eta)\sin^{\epsilon_2}(\omega) \\ a_3 \sin^{\epsilon_1}(\eta) \end{pmatrix} \tag{2}$$

where $\vec{X}$ is a three-dimensional vector containing the x, y and z components of the surface point. Superquadrics with $a_1 = a_2 = a_3 = 1$ correspond to the unit spheres in various metric spaces; that is, if $\epsilon_1 = \epsilon_2 = n$ then Equation 1 is the L_n distance metric. Such metric spaces have dual gradient spaces; thus the dual to Equation 2 is

$$\vec{N}(\eta,\omega) = \begin{pmatrix} \frac{1}{a_1}\cos^{2-\epsilon_1}(\eta)\cos^{2-\epsilon_2}(\omega) \\ \frac{1}{a_2}\cos^{2-\epsilon_1}(\eta)\sin^{2-\epsilon_2}(\omega) \\ \frac{1}{a_3}\sin^{2-\epsilon_1}(\eta) \end{pmatrix} \tag{3}$$

which is also the surface normal at the point (η,ω).

Note that because Equation 1 is a distance metric, it is trivial to determine where a point (x,y,z) is relative to the surface. One simply substitutes (x,y,z) into Equation 1; when the result is less than one the point is inside the surface, if greater then one the point is outside. The value d

$$d = ((x/a_1)^{2/\epsilon_1} + (y/a_2)^{2/\epsilon_1})^{\epsilon_1/\epsilon_2} + (z/a_3)^{2/\epsilon_2} - 1 \tag{4}$$

is a measure of how far a point has penetrated within the surface of the object. Similarly, it is easy to determine the surface normal at a collision point, by using Equation 2 to solve for (η,ω) and then using Equation 3 to calculate the surface normal. It is also straightforward to calculate the surface curvature at that point, and thus to characterize the shape of the contact patch. The ability to easily calculate these quantities is important for dynamic simulation, because it allows us to more accurately characterize contact between bodies than is possible within a standard FEM framework.

This basic building block specifies a *superset* of the standard constructive solid geometry (CSG) modeling primitives; it has been estimated that boolean combinations of such CSG primitives account for more than 80% of all manufactured parts. Many parts, however, cannot be described in this manner: they require the ability to modify or deform the basic superquadric-CSG shapes. Thus we must augment our basic vocabulary of shape with a set of *deformation operations*.

Thus, for instance, to model sheet metal one can use bending operations that mimic (at least roughly) real sheet-metal bends. It can be seen that by concatenating successive deformations together one can quickly model a very large space of shapes [1], just as by applying a sequence of manufacturing steps one can derive shapes very distant from the shape of the original stock. Thus one attractive characteristic of this representation scheme is that by concatenating successive deformations we can produce concise, natural descriptions of objects [1]. For situations involving a special family of shapes, such as turbine blades, one adds a deformation for each parameter of the shape family (e.g., blade curve, blade attack angle, blade tapering, etc.).

Deformations are calculated by mapping the original orthonormal space (in which the superquadric was defined) into a deformed space, e.g.,:

$$\vec{X}^* = \begin{pmatrix} 1 & 0 & k \\ 0 & 1 & k \\ 0 & 0 & 1 \end{pmatrix} \vec{X} \tag{5}$$

takes a form described by $\vec{X}$ and tapers it along the z axis. As a result of this deformation the surface normal $\vec{N}$ undergoes a similar transformation:

$$\vec{N}^* = \frac{1}{\det J}(J^{-1})^T \vec{N} \tag{6}$$

where J is the Jacobian of the deformation mapping [6].

Even with the addition of manufacturing-like deformations we still can't describe non-rigid dynamic behavior within the representational system, to describe how parts deform under acceleration, impact, or other stresses. In order to model such dynamic deformations we add deformations which correspond to the the objects's various *strain modes*.

When a object is subjected to a stress, it undergoes various deformations which can be calculated using a finite element method. However an alternative approach is by the method of *strain modes*, i.e., by linear superposition of the part's various vibration modes. (Nonlinear material properties can be accurately modeled by coupling the modes between each time step.) These modes are exactly the eigenvectors of the stress matrix calculated during finite element analysis [7,8]. Thus by augmenting our superquadric primitives with their strain modes we can produce analytic descriptions of dynamic systems consisting of these modeling primitives. The finite element method and modal analysis are further described and compared in the appendix.

In this approach one determines the deformation due to some force by calculating the coupling between that force and each of the deformation modes, and then changing each mode's amplitude parameter appropriately. Note that matrix inversion is not required. As more and more strain modes are added to the system, the answer obtained converges on that obtained by finite element analysis.

However normally the low-order strain modes — for example, the first and second order strain modes — that account for the vast majority of the shape variation a part experiences when subjected to stress. Thus if we calculate the deformation due to a force by adding up strain modes, we find that we can quickly achieve a quite good approximation using just the first few modes. To substantially improve this approximation, however, we have to add in a large number of high-order, low-amplitude strain modes. Thus while a good approximation can be achieved relatively quickly and cheaply, it requires a huge amount of additional calculation to obtain the last few decimal points of accuracy.

Thus our approach is to model only as many strain modes as are required. In a quick-and-dirty analysis — often sufficient during the exploratory phase of design — only rigid-body or rigid-body plus linear strain modes may be used. The result can be a huge savings in computation time. Later, more accurate analyses can add in more modes to achieve greater accuracy, although at greater cost.

We have found that most commonplace multi-body interactions can be adequately modeled by use of only rigid-body, linear, and quadratic strain modes [3], as is shown in Figure 4(b). Use of these modes for analysis was described in the Virtual Manufacturing scenario above.

To find these low-order deformation modes we can solve for the eigenvectors of the strain matrix; note that this operation typically only needs to be done when the static shape of the object is changed because only very small changes occur in low-order strain modes even when there is significant deformation due to loading. We have discovered, however, that, at least for visualization purposes, it is sufficient to use fixed, precomputed deformations modes that are parameterized only by the object's dimensions.

There are several computational advantages to this representational scheme. One advantage is that we can better characterize object-to-object contact because we can analytically describe surface curvature and surface normal. Perhaps most important, however, is the computational efficiency available with this scheme. In particular, rather than the combinatorially explosive scaling of the point-wise representation of FEM, we now have linear scaling ($kpm \log m$ vs $kp^2 m \log m$) for contact detection and constant (or no) scaling (k vs kn^2) when calculating object deformation. The result is that even for a small system with simple shapes (such as the example described above) our approach can be as much as two orders of magnitude more efficient than the standard approach. When dealing with complex situations where many objects interact, or when dealing with more complicated shapes, this efficiency advantage can become even larger. It is this scaling behavior that has allowed us to seriously attempt to build a Virtual Manufacturing environment.

6 SUMMARY

We have described an approach to interactively exploring the space of valid designs using computer simulation, a design paradigm we call Virtual Manufacturing. This approach to design attempts to integrate the interactive character of pencil and paper with the ability to quickly perform dynamic and structural analysis. We believe that such a design tool can fundamentally alter the nature of the design process in two separate ways. First by integrating functional analysis into the very first, exploratory stages of design, and second by recording the tests a design must pass together with the geometry of the design, thus allowing us to capture much of the design's functional semantics.

Finally, it must be emphasized that the current system is still very much a prototype; only enough of the system has been finished to allow a demonstration of feasibility and estimation of the execution times presented here. Future work includes considerable refinement of the dynamic analysis system, the constraint satisfaction system, and user interfaces. Two problems that will be the central in our near-term research are developing a better understanding of the trade-off between number of modes versus accuracy, and selection of particular real-world design problems where our system's performance might be evaluated.

[3] This is not true for bodies whose dimensions are quite disparate. However it is exactly these cases that can be reduced to one or two dimensions, and thus are cases in which standard FEM techniques are quite efficient

REFERENCES

[1]. Pentland, A. (1986) Perceptual Organization and the Representation of Natural Form, *Artificial Intelligence Journal*, Vol. 28, No. 2, pp. 1-38.

[2] Pentland, A., (1987) Towards and Ideal 3-D CAD System, *SPIE conference on Machine Vision and the Man-Machine Interface*, Jan. 11-16, San Diego, CA. SPIE Order No. 758-20.

[3] Sutherland, I., (1963), Sketchpad: A Man-Machine Graphical Communications System, in *1963 Spring Joint Computer Conference*, reprinted in Interactive Computer Graphics, H. Freeman, ed., IEEE Comp. Soc., 1980, pp. 1-19.

[4] Borning, A., (1979), Thinglab – a constraint-oriented simulation laboratory. SSL-79-3, Xerox PARC, Palo Alto, CA.

[5] Gardiner, M. (1965) The superellipse: a curve that lies between the ellipse and the rectangle, *Scientific American*, September 1965.

[6] Barr, A., (1984) Global and local deformations of solid primitives. *Computer Graphics 18*, 3, 21-30

[7] Williams, J., Musto, G., and Hawking, G., (1987) The Theoretical Basis of the Discrete Element Method, *Numerical Methods in Engineering, Theory, and Application*, Rotterdam: Balkema Publishers

[8] Williams, J. and Musto, G. (1987) Modal Methods for the Analysis of Discrete Systems, *Computers and Geotechnics*, Vol. 4, pp 1-19.

[9] Terzopolos, D., Platt, J., Barr, A., and Fleischer, K., (1987) Elastically deformable models, *Computer Graphics*, Vol. 21, No. 4, pp 205-214.

7 Appendix: Technical Details And Background

The finite element method (FEM) is a technique for simulating the dynamic behavior of an object. In the FEM the continuous variation of displacements throughout an object is replaced by a finite number of displacements at so-called nodal points. Displacements between nodal points are interpolated using a smooth function. Energy equations (or functionals) can then be derived in terms of the nodal unknowns and the resulting set of simultaneous equations can be iterated to solve for displacements as a function of impinging forces. In the dynamical case these equations may be written:

$$M\ddot{u} + D\dot{u} + Ku = f \tag{7}$$

where $\vec{u}$ is a $3n$ x 1 vector of the (x, y, z) displacements of the n nodal points relative to the objects' center of mass, M, D and K are $3n$ by $3n$ matrices describing the mass, damping, and material stiffness between each point within the body, and $\vec{f}$ is a $3n$ x 1 vector describing the (x, y, z) components of the forces acting on the nodes. This equation can be interpreted as assigning a certain mass to each nodal point and a certain material stiffness between nodal points, with damping being accounted for by dashpots attached between the nodal points. The damping matrix D is normally taken to be equal to $s_1 M + s_2 K$ for some scalars s_1, s_2. When $s_1 \neq 0$, $s_2 \neq 0$ this is called Raleigh damping, for $s_2 = 0$ it is called mass damping, and for $s_1 = 0$ stiffness damping.

To calculate the result of applying some force $\vec{f}$ to the object one discretizes the equations in time, picking an appropriately small time step, solves this equation for the new $\vec{u}$, and iterates until the system stabilizes. The FEM is most commonly used on single shapes with static loading. Relatively slowly-changing dynamic loading (e.g., changing wind, reciprocating motion) is somewhat less common. For these situations an implicit (direct) solution technique is most often used, which involves inverting large (e.g., larger than 100 x 100) matrices at each time step. Implicit solution is used because it allows much larger time steps; thus even though it is more expensive per time step it is less expensive overall [7]. Implict solution techniques are less desirable in the Virtual Manufacturing senario, because of the ability to take large time steps is limited by the presence of changing forces.

In such complex situations the use of explict, forward time-differencing techniques can be advantagous, e.g., [9]. Such methods involve multiplication, rather than inversion, of large matrices at each time step. Explict techniques can be viewed as performing a conjugate-gradient approximation of the implicit (direct) methods, so that it can be seen that many more time steps (order of the rank of the matrix) are required than solving using matrix inversion. When small time steps are required because of the complexity of the environment the efficiency of the explict method outweighs the disadvantage of smaller time steps.

7.1 Modal Analysis

Because M, D and K are normally positive definite symmetric, and M and D are assumed to be related by a scalar transformation, Equation 7 can be transformed into $3n$ independent differential equations by use of the *whitening transform*, which simultaneously diagonalizes M, D, and K. The whitening transform is the solution to the following eigenvalue problem:

$$\lambda\phi = M^{-1}K\phi \tag{8}$$

where $\bar{\lambda}$ and ϕ are the eigenvalues and eigenvectors of $M^{-1}K$.

Using the transformation $u = \phi\bar{u}$ we can re-write Equation 7 as follows:

$$\phi^T M\phi\ddot{\bar{u}} + \phi^T D\phi\dot{\bar{u}} + \phi^T K\phi\bar{u} = \phi^T f \quad . \tag{9}$$

In this equation $\phi^T M\phi$, $\phi^T D\phi$, and $\phi^T K\phi$ are diagonal matrices, so that if we let $\bar{M} = \phi^T M\phi$, $\bar{D} = \phi^T D\phi$, $\bar{K} = \phi^T K\phi$, and $\bar{f} = \phi^T f$ then we can write Equation 9 as $3n$ independent equations:

$$\bar{M}_i\ddot{\bar{u}}_i + \bar{D}_i\dot{\bar{u}}_i + \bar{K}_i\bar{u}_i = \bar{f}_i \quad , \tag{10}$$

where $\bar{M}_i$ is the i^{th} diagonal element of $\bar{M}$, and so forth. Because the modal representation diagonalizes these matrices it may be viewed as *preconditioning* the mass and stiffness matrices, with the attendant advantages of better convergence and numerical accuracy.

What Equation 10 describes is the time course of one of the object's *vibration modes*, hence the name *modal analysis* [8]. The constant $\bar{M}_i$ is the generalized mass of mode i, that is, it describes the inertia of this vibration mode. Similarly, $\bar{D}_i$, and $\bar{K}_i$ describe the damping and spring stiffness associated with mode i, and $\bar{f}_i$ is the amount of force coupled with this vibration mode. The i^{th} row of ϕ describes the *deformation* the object experiences as a consequence of the force $\bar{f}_i$, and the eigenvalue λ_i is proportional to the natural resonance frequency of that vibration mode.

Figure 4(b) illustrates the some of the first and second order modes of a cylinder. The first row of Figure 4(b) shows the cylinder at rest, the cylinder experiencing a linear deformation in response to a compressive force, and the cylinder experiencing a linear tapering deformation in response to an accelerating force. The second row shows a quadratic deformation in response to a centrally-applied (bending) force, and the next two illustrations show how both the linear and second order deformations can be superimposed to produce a more accurate simulation of the object's response to the compressive and accelerating forces than can be achieved by use of the linear modes alone.

To obtain an accurate simulation of the dynamics of an object one simply uses linear superposition of these modes to determine how the object responds to a given force. Because Equation 10 can be solved in closed form, we have the result that for objects composed of linearly-deforming materials *the non-rigid behavior of the object in response to an impulse force can be solved in closed form for any time t.* In environments with more complex forces, however, analytic solution becomes cumbersome and so numerical solution is preferred. Either explicit or implicit solution techniques may be used to calculate how each mode varies with time.

Non-linear materials may be modeled by summing the modes at the end of each time step to form the material *stress state* which can then be used to drive nonlinear plastic or viscous material behavior.

7.2 Number Of Modes Required

The modal representation decouples the degrees of freedom within the non-rigid dynamical system of Equation 7, but it does not reduced the total number of degrees of freedom. However once decoupled, we can separately analyze the various modes in order to determine which ones are required in order to obtain an accurate description of an object's non-rigid dynamic behavior.

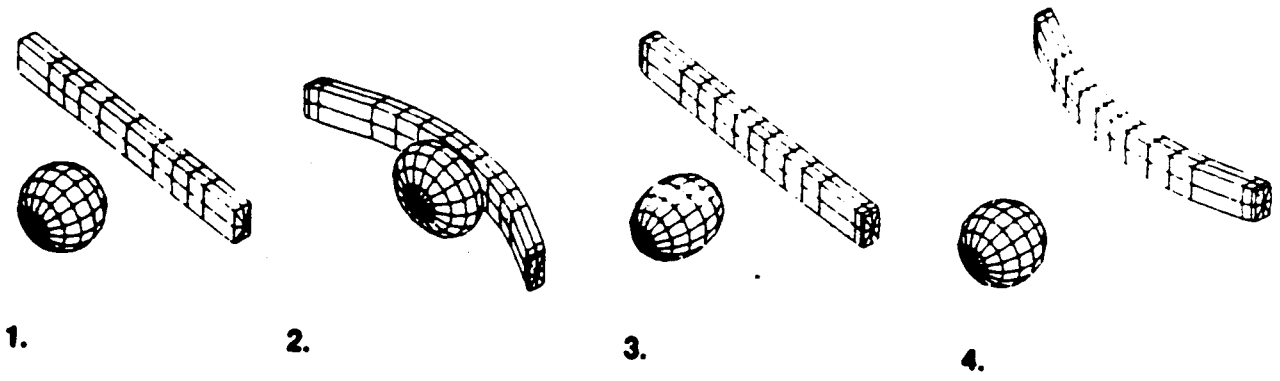


Figure 5: A ball colliding with a two-by-four

The most important observation is that modes associated with high resonance frequencies normally have little effect on object shape. This is because:

- The displacement amplitude for each mode is *inversely* proportional to the *square* of the mode's resonance frequency. Thus higher frequencies typically have small amplitudes.
- Damping is proportional to the mode's resonance frequency. Thus higher frequency vibrations dissipate quickly.
- Frequencies are excited in proportion to the frequency content of the input force. The force generated by simple collisions typically have a roughly Gaussian time course, so that an individual resonance frequency f receives energy proportional to $e^{-f^2/\sigma}$. Thus low frequencies typically receive more energy than high frequencies.

The combination of these effects is that high-frequency modes have very little amplitude, and even less effect, in simple collisions. As a consequence much more efficient (and still accurate) simulation of an object's dynamics can be accomplished by discarding the small, high-frequency modes, and considering only the large-amplitude, low-frequency modes. Exactly which modes to discard can determined by examining their associated eigenvalue, which determines the resonance frequency.

Experimentally, we have found that most commonplace multi-body interactions can be adequately modeled by use of only rigid-body, linear, and quadratic strain modes. Figure 5, for instance, shows a example of a simulated non-rigid dynamic interaction: a ball colliding with a two-by-four. As can be seen, the interaction and resulting deformations look realistic despite the use of only linear and quadratic modes.[4] Figure 4(b) also illustrates how use of linear and quadratic modes can accurately simulate non-rigid motion. Note, however, higher-order modes are required to accurately model the objects whose dimensions differ more than an order of magnitude.

7.2.1 Use of Fixed Modes

Normally, in either the finite element or modal methods, the mass, damping, and stiffness matrices are not recomputed at each time step. The use of fixed M, D, and K (or, equiv-

[4]Perhaps the most impressive fact about this example, however, is the speed of computation: Using a Symbolics 3600 (with a speed of roughly one MIP), it requires only one CPU second to compute each second of simulated time!

alently, fixed modes) is well-justified as long as the material displacements are small. The definition of "small," however, is quite different for different modes. Because the eigenvalue decomposition in Equation 8 performs a sort of principal-components analysis, it is the gross object shape (e.g., its low-order moments of inertia) determine the low-frequency modes, which as a consequence are quite stable. High-frequency modes are much less stable because they are determined by the fine features of the object's shape.

In the standard finite element formulation the action of each mode is distributed across the entire set of equations, so that one must recompute the mass and stiffness matrices as often as required by the very highest-frequency vibration modes. When these high-frequency modes are discarded the mass, damping, and stiffness matrices need to be recomputed much less frequently. In most situations it is sufficient to use single, fixed set of low-frequency modes throughout an entire simulation.

7.3 Combining Dynamics And Analytic Geometry

One problem with standard non-rigid dynamical techniques is that they are based on use of a point-wise representation of geometry, thus forcing the representation of geometry and dynamics to be identical. As a consequence one cannot, for instance, specify details of geometry without incurring large costs in calculating dynamic behavior, nor can one directly animate objects defined by, for example, large spline patches or constructive solid geometry. The fact that the same representation must be used for both geometry and dynamics thus has a large impact upon the efficiency and accuracy of multibody simulations, where detailed specification of geometry is required to obtain accurate detection and characterization of collisions.

We have been able to combine separate representations of dynamic behavior and geometric form in order to avoid these problems. We have accomplished this by describing each mode by an appropriate polynomial function, and then using global deformation techniques [6] to establish the correspondence between dynamic state and geometric state. The result is an efficient scheme for simulating non-rigid dynamics that can be applied in a unified manner to objects whose geometry is defined using a wide range of techniques.

To accomplish this, we first note that modes may be classified by the complexity of the associated deformation, e.g., as 0^{th} order (rigid body) modes, 1^{st} order (linear deformation) modes, 2^{nd} order (quadratic deformation) modes, and so forth, as was illustrated by Figure 4(b). Thus we can describe the deformation associated with each mode by use of polynomial deformation mappings of the appropriate degree. This is accomplished by performing a linear regression of a polynomial with m terms in appropriate powers of x, y, and z, against the n triples of x, y and z coefficients that compose ϕ_i, a $3n$ x 1 vector containing the elements of the i^{th} row of ϕ:

$$\alpha = (\beta^T \beta)^{-1} \beta^T \phi_i \quad , \tag{11}$$

where α is an m x 1 matrix of the coefficients of the desired deformation polynomial, β is an $3n$ x m matrix whose first column contains the elements of $\vec{p} = (x_1, y_1, z_1, x_2, y_2, z_2, ...)$, and whose remaining columns consist of the modified versions of $\vec{p}$ where the x, y, and/or

z components have been raised to the various powers, e.g.,

$$\beta = \begin{pmatrix} x_1 & x_1^2 & x_1 & x_1 & \cdots \\ y_1 & y_1 & y_1^2 & y_1 & \cdots \\ z_1 & z_1 & z_1 & z_1^2 & \cdots \\ x_2 & x_2^2 & x_2 & x_2 & \cdots \\ y_2 & y_2 & y_2^2 & y_2 & \cdots \\ z_2 & z_2 & z_2 & z_2^2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \end{pmatrix} . \tag{12}$$

The question of which polynomial powers are the appropriate for a particular column of ϕ can be decided either by inspection (noting that the order of the deformation is related to the associated eigenvalue), or automatically by including all combinations of powers of x, y, and z (up to some limit), performing the regression, and then discarding coefficients with negligible magnitude.

The result is a polynomial model of the unit amplitude deformation associated with mode i. By simply scaling this polynomial deformation according to the mode's amplitude we can accurately copy the effects of this mode on the object's shape. By superimposing these deformations we obtain an accurate accounting of the object's non-rigid deformation.